Package 'RCPmod'

September 29, 2017

Version 2.186

Date 2017-09-25

Title Regions of Common Profiles Modelling with Mixtures-of-Experts

Author Scott D. Foster

Maintainer Scott Foster <scott.foster@csiro.au>

Imports glmnet (>= 2.0-13), fishMod, MASS, gtools, parallel, stats, graphics

Description Identifies regions of common (species) profiles (RCPs), possibly when sampling artefacts are present. Within a region the probability of sampling all species remains approximately constant. This is performed using mixtures-of-experts models. The package also contains associated methods, such as diagnostics.

License GPL (>= 2)

SystemRequirements C++11

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-09-29 08:47:21 UTC

R topics documented:

`AIC.regimix` *Information criterion for a regimix model.*

Description

Returns information criterion for regimix models.

Arguments

`object`	an object obtained from fitting a region of common profile mixture model. Such as that generated from a call to regimix(qv).
`...`	ignored
`k`	the coefficient for the penalty in the information criterion. k=2 signifies Akaikes information criterion, k=log(object$n) corresponds to the Bayesian information criterion. If NULL (default) the AIC is used.

Value

A numeric scalar giving the Bayesian information criterion.

Method

AIC(object, ..., k=2)

BIC(object, ...)

Author(s)

Scott D. Foster

`coef.regimix` *A regimix objects coefficients.*

Description

Returns coefficicients from a regimix object.

Arguments

`object`	an object obtained from fitting a regions of common profile mixture model. Such as that generated from a call to regimix(qv).
`...`	ignored

Value

Returns a list of four elements, one each for the estimates for the species prevalence (alpha), the deviations from alpha for the first (nRCP-1) regional profiles (tau), the (nRCP-1) sets of region regression coefficents (beta), the coefficients for the species specific model (if specified in the model call), and the log of the dispersion parameter estimates (for negative binomial, Tweedie and Normal models).

Method

coef(object, ...)

Author(s)

Scott D. Foster

`cooks.distance.regimix`

Calculates leave-some-out statistics for a regimix object, principally a version of Cook's distance and cross-validated predictive logl

Description

Performs leave-some-out measures for a regimix model. This includes a measure of how much effect leaving out an observation has on the probability of each site's RCP label. Also, this function can be used as a cross-validation workhorse.

Arguments

model	a regmix object whose fit you want to assess
...	ignored
oosSize	the size of the withheld partitions (out-of-sample size). Use 1 (default) for leave-one-out statistics, such as Cook's distance and leave-one-out validation.
times	the number of times to perform the re-estimation (the number of leave out groups). For each 1:times a random partition of the data, of size oosSize, is taken and the model is fitted to one of the partitions. It is predicted to the other partition. The exception is when oosSize=1 and times=model$n (leave-one-out). In such cases (the default too), the observations are left out one-by-one and not randomly.
mc.cores	the number of cores to spread the workload over. Default is 1. Argument is useless on Windows machines – see `?parallel::mclapply`
quiet	should printing be suppressed? Default is no, it should not. Note that in either case, printing of the iteration trace etc is suppressed for each regimix fit.

Value

An object of class `regiCooksD`. It is a list of 4 elements:

Y	the species data,
CV	the model$n by model$S by times array of out-of-sample predictions (this array contains a lot of NAs for where predictions would in-sample),
cooksD	a model$n by model$nRCP matrix of statistics that resemble Cook's distance. The statistic is the change in the prediction of RCP probability from the model with all the data to the model with only the in-sample data, and
predLogL	the predictive log-likelihood of each point in each withheld sample (log-likelihood contributions of withheld observations, again there will be many NAs).

Method

cooks.distance(model, ..., oosSize = 1, times = model$n, mc.cores = 1, quiet = FALSE, type = "cooksD")

See Also

regimix stability.regimix

Examples

```
## Not run:
#not run as R CMD check complains about the time taken.
#This code will take a little while to run (<1 minute on my computer)
#For leave-one-out cooks distance, use oosSize=1
#for serious use times will need to be larger.
system.time({
example( regimix);
```

```
cooksD <- cooks.distance( fm, oosSize=10, times=25)
})
example( regimix)  #will fit a model and store in fm
#for serious use times will need to be larger.
#For leave-one-out cooks distance, use oosSize=1
cooksD <- cooks.distance( fm, oosSize=10, times=5)

## End(Not run)
```

`extractAIC.regimix` *Extracts the AIC for a regimix model.*

Description

Computes the generalised AIC for a regimix model.

Arguments

`object`	an x obtained from fitting a region of common profile mixture model. Such as that generated from a call to regimix(qv).
`scale`	ignored
`k`	the coefficient for the penalty in the information criterion. k=2 corresponds to Akaikes information criterion, k=log(x$n) corresponds to the Bayesian information criterion. Default is k=2 (AIC).
`...`	ignored

Value

A two element numeric vector. First element is the number of parameters in the model. The second is the information criterion.

Method

extractAIC(object, scale, k=2, ...)

Author(s)

Scott D. Foster

`logLik.regimix` *The log likelihood for a regimix model.*

Description

Returns the maximised log likelihood for a regimix model.

Arguments

`object`	an object obtained from fitting a regitat mixture model. Such as that generated from a call to regimix(qv).
...	ignored

Value

A numeric scalar giving the maximised log likelihood for the regimix model.

Method

logLik(object, ...)

Author(s)

Scott D. Foster

`orderFitted` *Assesses classification error and re-orders parameters for simulated data (orderFitted) or for two fits (orderPost).*

Description

Finds the set of (posterior) group labels that most closely matches the simulated data (orderFitted) or another model fit (orderPost). This function is not intended to be used by the general public – please treat with care. The only help given is the code itself. This is the ultimate R experience.

Author(s)

Scott D. Foster

`plot.regimix` *Plots residuals from a regimix x.*

Description

Plots the residuals from a regimix x (after calculating them).

Arguments

`x`	an x obtained from fitting a RCP mixture model. Such as that generated from a call to regimix(qv).
`type`	the type of residual to be plotted. Options are the default "RQR" for randomised quantile residuals (see Dunn and Smyth (1996) and Foster et al (in prep) for details) and "deviance" for the square root of minus two times the log likelihood contributions for each site (see Foster et al, 2013).
`nsim`	for type=="RQR" this argument is ignored. For type=="deviance" gives the number of simulations to use for the confidence interval. The default is 100, serious usage is likely to require more.
`alpha.conf`	the confidence level(s) to use in the residual plots for type=="deviance". Default is c(0.90,0.95,0.99). Ignored if type=="RQR."
`quiet`	should printing be performed? quiet=FALSE (default) says yes!
`species`	which species should be included in the residual plot. Default is "AllSpecies" and any subset of these (see x$names$spp for those available) is accepted.
`fitted.scale`	which scale to plot the fitted values on the x-axis? Options are "response" (default), "log" (useful sometimes for log-link models), and "logit" (useful sometimes for logit-link models).
`...`	ignored

Details

The two types of residuals are inherently different. The "RQR" residuals produce a residual for each species at each site and the "deviance" residuals produce a site residual (no species level residual). The plots also differ, the "RQR" type generates a single normal QQ-plot for all species and all sites, and a residual versus fitted plot for all species and sites (Described in Foster et al, 2013). The "deviance" type generates a pair of Tukey mean-difference plots, similar in spirit to a QQ-plot. The first is for point-wise confidence intervals and the second is for approximate global intervals. See Foster et al (2013) for details.

The distribution for the "RQR" residuals should be standard normal. For "deviance" residuals, the distribution is unknown and simulation is used to graphically assess how odd the observed residuals look compared to ones generated assuming the model is correct.

Method

plot(x, ..., type="RQR", nsim = 100, alpha.conf = c(0.9, 0.95, 0.99), quiet=FALSE)

Author(s)

Scott D. Foster

References

Dunn, P.K. and Smyth G.K. (1996) Randomized Quantile Residuals. Journal of Computational and Graphical Statistics *5*: 236–244.

Foster, S.D., Givens, G.H., Dornan, G.J., Dunstan, P.K. and Darnell, R. (2013) Modelling Regions of Common Profiles Using Biological and Environmental Data. Environmetrics *24*: 489–499. DOI: 10.1002/env.2245

Foster, S.D., Lyons, M. and Hill, N. (in prep.) Ecological Groupings of Sample Sites in the presence of sampling artefacts.

`plot.registab` *Diagnostic plotting to see if RCP groups are stable*

Description

For increasing size of hold-out samples, cooks distance and predictive log-likelihood are plotted.

Arguments

	You can put normal text in **Arguments**, too, like this. Remember to indent all arguments, as below.
	a registab object (the output of stability.regimix)
`x`	ignored.
`minWidth`	the minimum width of the density for each hold-out size. A rectangle of minWidth (at least) will be placed, and centered, at each of the hold-out sizes.
`ncuts`	number of cuts to use to describe the distributions of predicted log-likelihood. This number is a maximum value and the actual number used will depend on the data (see the breaks argument of `hist`)
`ylimmo`	y-limits on the predicted log-likelihood plot. Typically, upper bound will be zero. If NULL (default), the lower bound is taken to be the smallest observation in the data.
`...`	ignored

Value

Nothing, but causes plots to be produced on the graphics device

Method

plot.registab(x, y, minWidth=1, ncuts=111, ylimmo, ...)

See Also

stability.regimix, regimix, cooks.distance.regimix

Examples

```
## Not run:
#not run as R CMD check complains about the time taken.
#This code will take a little while to run (about 3.5minutes on my computer)
system.time({
example( regimix);
my.registab <- stability.regimix( fm, oosSizeRange=seq( from=1,to=fm$n%/%5,length=5),
      times=fm$n, mc.cores=2, doPlot=FALSE);
plot( my.registab, minWidth=1, ncuts=15);
})

## End(Not run)
```

---

predict.regimix *Predicts from a regimix object.*

Description

Predicts RCP probabilities at a series of sites. Confidence intervals are available too.

Arguments

object	an object obtained from fitting a RCP mixture model. Such as that generated from a call to regimix(qv).
object2	a regiboot object obtained from bootstrapping the regimix object. Such as that generated from a call to regiboot(qv). If not supplied, then predict.regimix will do parametric bootstrapping (otherwise non-parametric bootstrap).
newdata	a data.frame (or something that can be coerced) containing the values of the covariates where predictions are to be made. If NULL (the default) then predictions are made at the locations of the original data.
nboot	the number of parametric bootstrap samples to take for the bootstrap predictions, standard errors and confidence intervals. The default is 0, that is no bootstrapping is to be done and point predictions only are given. If object2 is not NULL, then the number of bootstrap samples is taken from that object (this argument is then ignored).
alpha	a numeric within [0,1] (well [0.5,1] really) indicating the specified confidence for the confidence interval. Argument is redundant if nboot == 0.
mc.cores	the number of cores to spread the computations over. Ignored if running on a Windows machine.
...	ignored

Details

This function implements two separate, and quite different, bootstrapping routines. The first, attributable to Foster et al (2013), which implements a *parametric* bootstrap, whereby parameters are drawn from their sampling distribution (defined by the ML estimates and their asymptotic vcov matrix). Yes, the vcov function needs to be run first and stored in the the regimix object as $vcov. Typically, the vcov matrix is obtained using numerical derivatives, which can be slow to calculate and somewhat unstable/erratic. This was the original suggestion and has been superceeded by the *non-parametric* bootstrap routine. This is described in Foster et al (in prep) and bootstraps the sampling site data repeatedly, and for each bootstrap sample the model is re-estimated. Variation in the bootstrap samples is carried forward to the prediction step to guage the uncertainty.

The parametric bootsrap implementation of this function can take a while to run – it is a bootstrap function. nboot samples of the parameters are taken and then used to predict at each set of covariates defined in newdata. Quantiles of the resulting sets of bootstrap predictions are then taken. It is the last step that really takes a while. The non-parametric version of this function should not take as long as the grunt work of bootstrapping is carried out in the regiboot(qv) function.

Note that this function is not implemented. It could be, using the parallel package, but it is currently not. The bulk of the bootstrap calculations are done in C++, which reduces the waiting time but parallelising it would be even better.

Value

If nboot==0 then a n x H matrix of prior predictions (n=nrow(newdata), H=number of RCPs). Each row should sum to one.

`if nboot!=0 then a list is returned. It has elements:`

`ptPreds`	the n x H matrix of point predictions
`bootPreds`	the n x H matrix of bootstrap point predictions (mean of bootstrap samples)
`bootSEs`	the n x H matrix of bootstrap standard errors for predictions
`bootCIs`	the n x H x 2 array of bootstrap confidence intervals. Note that bootCIs[,,1] gives the lower CIs and bootCIs[,,2] gives the upper CIs.

Method

predict(object, object2=NULL, ..., newdata=NULL, nboot=0, alpha=0.95, mc.cores=1)

Author(s)

Scott D. Foster

print.regimix *Prints a regimix object.*

Description

Prints some attributes of a regimix object.

Arguments

x	an object obtained from fitting a RCP mixture model. Such as that generated from a call to regimix(qv).
...	ignored

Details

A list is returned that will be printed on exit, if not assigned to anything. It contains the function call and the estimated coefficients.

Method

print(x, ...)

Author(s)

Scott D. Foster

regiboot *Bootstraps a regimix object.*

Description

Performs bootstrap sample and estimation for regimix objects. Useful for calculating measures of uncertainty in predictions from a regimix object, and also about the regimix parameter estimates. This function can be used in conjunction with vcov.regimix(qv) and predict.regimix(qv). In partciular, these bootstrap samples can be used to gauge variability in parameter estimates and hence the model itself.

Usage

```
regiboot( object, nboot=1000, type="BayesBoot", mc.cores=1, quiet=FALSE,
                                                orderSamps=FALSE, MLstart=TRUE)
```

Arguments

object	an object obtained from fitting a RCP mixture model (class "regimix"). Such as that generated from a call to regimix(qv). This object will need to be created with the argument titbits=TRUE as these pieces of data are needed in the re-fitting. Of course, you could try to just save parts of titbits but the memory-hungry data is all required.
nboot	a numeric scalar giving the number of bootstrap samples to obtain. More is better, but takes longer.
type	a character string giving the type of bootstrap to perform. Options are:"SimpleBoot" which gives sample resampling, and "BayesBoot" (default) which gives Bayesian Bootstrap sampling. The nomenclature, and the Bayesian Bootstrap, come from Rubin (1981).
mc.cores	an integer giving the number of cores to run the bootstrap samples on. The default is 1, that is no parallelisation. This parameter is redundant on Windows machines as the method of parallelisation, mclapply(), is not available there.
quiet	should the progress bar be printed to the output device?If quiet=FALSE (default) then the progress bar is printed.
orderSamps	should each bootstrap sample be re-ordered to that permutation of RCP ordering that best matches the initial model? Default FALSE implies no re-ordering.
MLstart	should each bootstrap estimation start at the original model's ML estimate? Default is TRUE for yes it should.

Details

This function can take a while to run – it is a bootstrap function. nboot re-samples of the data are taken and then the parameters are estimated for each re-sample. The function allows for parallel calculations, via mclapply(qv), which reduces some of the computational burden. To use parallel computing, specify mc.cores>1. Note that this will not work on Windows computers, as mclapply(qv) will not work.

The Bayesian bootstrap method is operationally equivalent to the simple bootstrap, except that the weightings are non-integral. See Rubin (1981).

It might be tempting to reduce the tolerance for convergence of each estimation procedure. We recommend not doing this as it is likely to have the effect of artificially reducing estimates of uncertainty. This occurs as the resampled estimates are liekly to be closer to their starting values (the MLEs from the original data set).

Value

An object of class "regiboot", which is essentially a matrix with nboot rows and the number of columns equal to the number of parameters matrix. Each row gives a bootstrap estimate of the parameters.

Author(s)

Scott D. Foster

References

Foster, S.D., Lyons, M. and Hill, N. (in prep.) Ecological Groupings of Sample Sites in the presence of sampling artefacts.

Rubin, D.B. (1981) The Bayesian Bootstrap. The Annals of Statistics *9*:130–134.

`regimix` *Fits a regimix model.*

Description

Fits a mixture-of-experts model to identify regions of similar community composition.

Usage

```
regimix( form.RCP = NULL, form.spp = NULL, data, nRCP = 3, dist="Bernoulli",
   offset=NULL, weights=NULL, control = list(), inits = "random2",
   titbits = TRUE, power=1.6)
regimix.multifit( form.RCP = NULL, form.spp = NULL, data, nRCP = 3, dist="Bernoulli",
   offset=NULL, weights=NULL, control = list(), inits = "random2",
   titbits = FALSE, power=1.6, nstart=10, mc.cores=1)
```

Arguments

`form.RCP`	an object of class "formula" (or an object that can be coerced to that class). The left hand side of this formula specifies the columns of the data argument that are the species binary data. The right hind side of this formula specifies the dependence of the `region of common profile` (RCP) probabilities on covariates. An example formula is cbind(spp1,spp2,spp3)~1+poly(temp,3) where spp1, spp2 and spp3 are species labels (columns of data), and the RCP probabilities depend on a (link-)cubic polynomial of temp.
`form.spp`	an object of class "formula" (or an object that can be coerced to that class). The left hand side of this formula should be left empty (it is removed if it is not empty). The right hand side of this formula specifies the dependence of the species"'" data on covariates (typically different covariates to form.RCP to avoid confusing confounding). An example formula is ~gearType + timeOfDay, where gearType describes the different sampling gears and timeOfDay describes the time of the sample. Any intercept term in this model is removed (as this is dealt with in the species-specific RCP-specific intercepts).
`data`	an object of class "data.frame" (or one that can be coerced to that class). This data.frame has to contain all the data for the terms in the form.RCP, form.spp and offset arguments.
`nRCP`	an integer (or something that can be coerced to an integer). This argument specifies the number of RCPs that will be fitted.

`dist` a character string describing the distribution of the species data. Must be one of "Bernoulli" (for binary data), "Poisson" and "NegBin" (for count data), "Tweedie" (for biomass, non-negative, data), and "Normal" (for quantity data). Note that this also specifies the link function used for the data. The links are:logit (for "Bernoulli"), log (for "Poisson", "NegBin" and "Tweedie") and identity (for "Normal"). Note that the computation strategy for the Tweedie model is slightly different than for the other models. The Tweedie models can also take a while to fit, even for small data sets, as calculating Tweedie densities can be quite labourious.

`offset` a numeric vector of length nrow(data) that is included into the model as an offset. It is included into the conditional part of the model where conditioning is performed on the unobserved RCP type. Note that offsets cannot be included as part of the form.RCP or form.spp arguments – only through this argument.

`weights` a numeric vector of length nrow(data) that is used as weights in the log-likelihood calculations. If NULL (default) then all weights are assumed to be identically 1.

`control` a list of control parameters for optimisation and calculation. See details.

`inits` either a character string or a numeric vector. If a character string ("random2", "random", "hclust" or "noPreClust") then it gives the method to generate starting values. If a numeric vector then it specifies the values of alpha, tau, beta, gamma, and log(dispersions), in that order. It will be used unchecked if not a character. The default is "random2" which is described in Foster et al (in prep.). Other choices are "random", which is described in Foster et al (2013), "hclust" which is the same as "random2" but with no random component (also described in Foster et al (2013)). The "noPreClust" choice is designed for situations where running hclust() on the data is not feasible (due to data size) or not wanted – instead the initial (design) matrix for assigning sites to RCPs is obtained by samples from a symmetric Dirichlet distribution with shape parameter 5. It is strongly advised that multiple starts with multiple (random) start locations are performed. The reason for this is that the log-likelihood surface can be fairly "bumpy" with multiple local maxima. Multiple starts guards somewhat against making inference from these local maximia. For regimix.multifit, inits should be the either "random" or "random2" (default).

`titbits` either a boolean or a vector of characters. If TRUE (default for regimix(qv)), then some objects used in the estimation of the model"'"s parameters are returned in a list entitled "titbits" in the model object. Some functions, for example plot.regimix(qv) and predict.regimix(qv), will require some or all of these pieces of information. If titbits=FALSE (default for regimix.multifit(qv)), then an empty list is returned. If a character vector, then just those objects are returned. Possible values are:"Y" for the outcome matrix, "X" for the model matrix for the RCP model, "W" for the model matrix for the species-specific model, "offset" for the offset in the model, "wts" for the model weights, "form.RCP" for the formula for the RCPs, "form.spp" for the formula for the species-specific model, "control" for the control arguments used in model fitting, "dist" for the conditional distribution of the species data, and "power" for the power parameters used (only used in Tweedie models). Care needs to be taken when using titbits=TRUE in regimix.multifit(qv) calls as titbits is created for EACH OF THE MODEL FITS. If the data is large or if nstart is large, then setting titbits=TRUE

may give users problems with memory. It is more efficient, from a memory perspective, to refit the "best" model using regimix(qv) after identifying it with regimix.multifit(qv). See examples for illustration about how to do this.

`power` a numeric vector (length either 1 or the number of species) defining the power parameter to use in the Tweedie models. If length(power)==1, then the same power parameter is used for all species. If length(power)==No_species, then each species gets its own power parameter. Power values must be between 1 and 2, for computational reasons they should be well away from the boundary. The default is 1.6 as this has proved to be a good ball-park value for the fisheries data that the developer has previously analysed.

`nstart` for regimix.multifit only. The number of random starts to perform for re-fitting. Default is 10, which will need increasing for serious use.

`mc.cores` for regimix.multifit only. The number of cores to spread the re-fitting over.

Details

A typical formula for use in the form.RCP argument will have the form (for example) cbind(spp1,spp2,spp3,spp4)~1+cov1+co This signifies that there are 4 species to be used for RCP modelling and that the RCP types are dependent on cov1+cov2+cov3+cov2:cov3. See ?glm for a description of how the right hand side of the formula is expanded.

Likewise a typical formula for use in the form.spp argument will have the form (for example) ~1+fac1+cov1. This signifies that the catchabilities of each species depends upon the levels of the factor fac1 and the covariate cov1. See ?glm for a description of how the right hand side of the formula is expanded.

The computation strategy for the default method, which has been demonstrated to work for all data sets the developers have encountered thus far, is fully described in Foster et al (2013) and Foster et al (2017). We note however, that it is a good idea to standardise covariates prior to calling regimix. This is not formally required by the model, but it does drastically reduce the chance of numerical issues in the first iteration. If you choose to NOT standardise, then you should at least choose a scale that is reasonable (so that the numerical range is measured by units and not thousands of units). This may mean that the units may be, for example, kilometres (and not metres), or 100s of kilometres (and not metres/kilometres).

We do not, on purpose, provide residuals as a routine part of the model. Users should use the residuals.regimix(qv) function to obtain them. We do this as the type of residual needs to be specified (although we recommend type=="RQR" for routine use).

Control arguments for optimisation generally follow those in optim(qv), although a few differences occur (e.g. "loglOnly"). The elements of the control list are

maxit The maximum number of iterations. Default is 500.

quiet Should any reporting be performed? Default is FALSE, for reporting. For regimix.multifit(), this indicates if the progress should be printed.

trace Non-negative integer. If positive, tracing information on the progress of the optimization is produced. Higher values may produce more tracing information.

nreport The frequency of reports for optimisation. Default is 10 – a report for 10th iteration.

reltol Relative convergence tolerance. The algorithm stops if it is unable to reduce the value by a factor of reltol * (abs(val) + reltol) at a step. Defaults to sqrt(.Machine$double.eps), typically about 1e-8.

optimise Should optimisation for estimation occur? If TRUE (default) optimisation will occur. If FALSE no optimisation is performed.

loglOnly Should the log-likelihood be caulcated? If TRUE (default) then log-likelihood is calculated and returned. If FALSE then the log-likelihood is not calculated for return.

derivOnly Should the scores be evaluated at the (final) parameter values. If TRUE (default) then they are calculated. If FALSE then they are not calculated.

penalty A numeric scalar. This is the concentration for the Dirichlet-inspired penalty for the prior probabilities. Values less than zero will be set to the default (0.1). Large values give more penalisation than small ones.

penalty.tau A numeric scalar. This is the penalty for the tau parameters in the species model. They are assumed to come from a normal distribution with standard deviation given as this parameter (default is 10).

penalty.gamma A numeric scalar. This is the penalty for the gamma parameters in the species model. They are assumed to come from a normal distribution with standard deviation given as this parameter (default is 10).

penalty.disp a two element vector. These are combined to form the penalty for the dispersion parameters (if any). The dispersions are assumed to come from a log-normal distribution with log-mean penalty.disp[1] and log-standard-deviation penalty.disp[2]. Defaults to c(10,sqrt(10)), which gives shrinkage towards 1 (the mode of the penalty). Note that for Normal models, where the dispersion alone defines the variance, a strong penalty may be required to keep parameters estimable.

For calls to regimix.multifit(), titbits is set to FALSE– so no excess memory is used. If users want this information, and there is good reason to want it, then a call to regimix() with starting values given as the `best` fit's estimates should be used.

Value

regimix returns an object of class `regimix` and regimix.multfit returns a list of objects of class `regimix`. The regimix class has several methods: coef, plot, predict, residuals, summary, and vcov. The regimix object consists of a list with the following elements:

`AIC`	Akaike an information criterion for the maximised model.
`BIC`	Bayesian information criterion for the maximised model.
`call`	the call to the function.
`coefs`	a list of three elements, one each for the estimates for the species prevalence (alpha), the deviations from alpha for the first (nRCP-1) RCP (tau), and the (nRCP-1) sets of RCP regression coefficents (beta).
`conv`	the convergence code from the maximisation procedure. See ?optim for an explanation (basically 0 is good and anything else is bad).
`dist`	the character string identifying the distribution used for the model.
`logCondDens`	an nObs by nRCP matrix specifying the probability of observing each sites"'" data, given each of the RCP types.

logl	the maximised log likelihood.
mus	an array of size nRCP x S x nRCP where each element of the first dimension is the fitted value for all the species in all the RCP types.
n	the number of samples.
names	the names of the species, and the names of the covariates for X and W.
nRCP	the number of RCPs.
pis	an n x nRCP matrix with each column giving the prior probabilities for the corresponding RCP type. Rows sum to one.
postProbs	an n x nRCP matrix with each column giving the posterior probabilities for the corresponding RCP type. Rows sum to one (as each site is assumed to be from one of the RCP types).
p.w	the number of covariates used in the species-specific model.
p.x	the number of covariates used in the RCP model
S	the number of species.
scores	a list of three elements. Structure corresponds to coefs.
start.vals	the values used to start the estimation procedure.
titbits	(if requested using the titbit argument, see above) other pieces of information, useful to developers, that users should not typically need to concern themselves with. However, this information is used by methods for regimix objects.

Author(s)

Scott D. Foster

References

Foster, S.D., Givens, G.H., Dornan, G.J., Dunstan, P.K. and Darnell, R. (2013) Modelling Regions of Common Profiles Using Biological and Environmental Data. Environmetrics *24*: 489–499. DOI: 10.1002/env.2245

Foster, S.D., Lyons, M. and Hill, N. (2017) Ecological Groupings of Sample Sites in the presence of sampling artefacts. Journal of the Royal Statistical Society – Series C *XX*: XX–XX. DOI: 10.1111/rssc.12211

Examples

```
#simulate data
example( simRCPdata)  #generates Negative Binomial data
#fit the model
my.form.RCP <- paste( paste( paste(
  'cbind(', paste( paste( 'spp', 1:S, sep=''), collapse=','), sep=''),
    ')',sep=''),
       '~x1.1+x1.2+x1.3+x2.1+x2.2+x2.3',sep='')
my.form.spp <- ~w.1+w.2+w.3
fm <- regimix( form.RCP = my.form.RCP, form.spp=my.form.spp, data = simDat,
                 dist="NegBin", nRCP = 3, inits = "random2", offset=offset)
## Not run:
```

```
#fit the model using multiple starting values
fm <- regimix.multifit( form.RCP = my.form.RCP, form.spp=my.form.spp, data = simDat,
  dist="NegBin", nRCP = 3, inits = "random2", offset=offset, nstart=10, titbits=FALSE,
  mc.cores=1)
#sometimes the model 'mis-fits' and one or more of the RCP groups has no sites associated
#with it.  These need to be removed (based on the colSums of the posterior probabilities)
postProbSums <- t( sapply( fm, function(x) colSums( x$postProbs)))
#Identify those models with zero posterior prob classes
allGoodUns <- apply( postProbSums, 1, function(x) all(x!=0))
#subset the fits
fm.clean <- fm[allGoodUns]
#choose the model with the lowest BIC
goodUn <- which.min( sapply( fm.clean, BIC))
#Using the 'best' model, use regimix(qv) again to additional model output needed for other
#functions (e.g. plot.regimix(qv), predict.regimix(qv) and regiboot(qv)). Note that the
#model is not estimated again (see control argument of the following regimix(qv) call.
fm.final <- regimix( form.RCP = my.form.RCP, form.spp=my.form.spp, data = simDat,
  dist="NegBin", nRCP = 3, inits = unlist( fm.clean[[goodUn]]$coef),
  control=list(optimise=FALSE), offset=offset)

## End(Not run)
```

`residuals.regimix` *Residuals for a regimix object.*

Description

Returns the (absolute) deviance residuals or the randomised quantile residuals from a regimix object.

Arguments

`object`	an object obtained from fitting a RCP mixture model. Such as that generated from a call to regimix(qv).
`...`	ignored
`type`	either "RQR" (the default) or "deviance" (the old default). See details.
`quiet`	should information be printed during the function? quiet=FALSE gives the information, whereas quiet=TRUE does not.

Details

The randomised quantile residuals ("RQR", from Dunn and Smyth, 1996) are defined by their marginal distribution function (marginality is over other species observations within that site; see Foster et al, in prep). The result is one residual per species per site and they all should be standard normal variates. Within a site they are likely to be correlated (as they share a common latent factor), but across sampling locations they will be independent.

The deviance residuals (as used here), are actually just square root of minus two times the log-likelihood contribution for each sampling location. We do not subtract the log-likelihood of the

saturated model as, at the time of writing, we are unsure what this log-likelihood should be (latent factors confuse things here). This implies that the residuals will not have mean zero and their variance might also be heteroskedastic. This was not realised when writing the original RCP paper (Foster et al, 2013), obviously. We still believe that these residuals have some utility, but we are unsure where that utility stops. For general useage, the "RQR" residuals should probably be preferred.

Value

```
For type=="RQR", a number-of-sites by number-of-species matrix with the randomised quantile residu

For type=="deviance" a numeric vector of size object$n containing the deviance residuals.
```

Method

residuals(object, ..., type="RQR", quiet=FALSE)

Author(s)

Scott D. Foster

References

Dunn, P.K. and Smyth G.K. (1996) Randomized Quantile Residuals. Journal of Computational and Graphical Statistics *5*: 236–244.

Foster, S.D., Givens, G.H., Dornan, G.J., Dunstan, P.K. and Darnell, R. (2013) Modelling Regions of Common Profiles Using Biological and Environmental Data. Environmetrics *24*: 489–499. DOI: 10.1002/env.2245

Foster, S.D., Lyons, M. and Hill, N. (in prep.) Ecological Groupings of Sample Sites in the presence of sampling artefacts.

---

simRCPdata *Simulates from a regimix model*

Description

Simulates a data set from a mixture-of-experts model for RCP (for region of common profile) types.

Usage

```
simRCPdata(nRCP=3, S=20, n=200, p.x=3, p.w=0, alpha=NULL, tau=NULL, beta=NULL,
                        gamma=NULL, logDisps=NULL, powers=NULL, X=NULL, W=NULL,
                                                offset=NULL, dist="Bernoulli")
```

Arguments

nRCP	Integer giving the number of RCPs
S	Integer giving the number of species
n	Integer giving the number of observations (sites)
p.x	Integer giving the number of covariates (including the intercept) for the model for the latent RCP types
p.w	Integer giving the number of covariates (excluding the intercept) for the model for the species data
alpha	Numeric vector of length S. Specifies the mean prevalence for each species, on the logit scale
tau	Numeric matrix of dimension c(nRCP-1,S). Specifies each species difference from the mean to each RCPs mean for the first nRCP-1 RCPs. The last RCP means are calculated using the sum-to-zero constraints
beta	Numeric matrix of dimension c(nRCP-1,p.x). Specifies the RCP's dependence on the covariates (in X)
gamma	Numeric matrix of dimension c(n,p.w). Specifies the species' dependence on the covariates (in W)
logDisps	Logartihm of the (over-)dispersion parameters for each species for negative binomial, Tweedie and Normal models
powers	Power parameters for each species for Tweedie model
X	Numeric matrix of dimension c(n,p.x). Specifies the covariates for the RCP model. Must include the intercept, if one is wanted. Default is random numbers in a matrix of the right size.
W	Numeric matrix of dimension c(n,p.w). Specifies the covariates for the species model. Must *not* include the intercept. Unless you want it included twice. Default is to give random levels of a two-level factor.
offset	Numeric vector of size n. Specifies any offset to be included into the species level model.
dist	Text string. Specifies the distribution of the species data. Current options are "Bernoulli" (default), "Poisson", "NegBin", "Tweedie" and "Normal".

Value

A data frame that contains the outcomes (species data) and the covariates (environmental data and species-level covariates). This data.frame has a number of special attirbutes, which are information about the model underlying the data. They are:

RCPs	the true, but unobserved, RCP types
pis	the true prior probabilities
alpha	the species overall prevalences, on linear predictor scale
tau	the deviation from alpha for each RCP type, on linear predictor scale
beta	the parameters controlling how the RCP types depend on the covariates

gamma	the parameters controlling how each species depends on the species-level covariates
logDisps	the logarithm of the dispersion parameter for each species
mu	the probabilities of each species occuring in each RCP type

Author(s)

Scott D. Foster

References

Foster, S.D., Givens, G.H., Dornan, G.J., Dunstan, P.K. and Darnell, R. (2013) Modelling Regions of Common Profiles Using Biological and Environmental Data. Environmetrics.

Examples

```
#generates synthetic data
set.seed( 151)
n <- 100
S <- 10
nRCP <- 3
my.dist <- "NegBin"
X <- as.data.frame( cbind( x1=runif( n, min=-10, max=10), x2=runif( n, min=-10, max=10)))
Offy <- log( runif( n, min=30, max=60))
pols <- list()
pols[[1]] <- poly( X$x1, degree=3)
#important to scale covariates so that regimix can get half-way decent starting values
pols[[2]] <- poly( X$x2, degree=3)
X <- as.matrix( cbind( 1, X, pols[[1]], pols[[2]]))
colnames( X) <- c("const", 'x1', 'x2', paste( "x1",1:3,sep='.'), paste( "x2",1:3,sep='.'))
p.x <- ncol( X[,-(2:3)])
p.w <- 3
W <- matrix(sample( c(0,1), size=(n*p.w), replace=TRUE), nrow=n, ncol=p.w)
colnames( W) <- paste( "w",1:3,sep=".")
alpha <- rnorm( S)
tau.var <- 0.5
b <- sqrt( tau.var/2)
#a double exponential for RCP effects
tau <- matrix( rexp( n=(nRCP-1)*S, rate=1/b) - rexp( n=(nRCP-1)*S, rate=1/b), nrow=nRCP-1, ncol=S)
beta <- 0.2 * matrix( c(-1.2, -2.6, 0.2, -23.4, -16.7, -18.7, -59.2, -76.0, -14.2, -28.3,
  -36.8, -17.8, -92.9,-2.7), nrow=nRCP-1, ncol=p.x)
gamma <- matrix( rnorm( S*p.w), ncol=p.w, nrow=S)
logDisp <- log( rexp( S, 1))
set.seed(121)
simDat <- simRCPdata( nRCP=nRCP, S=S, p.x=p.x, p.w=p.w, n=n, alpha=alpha, tau=tau,
  beta=beta, gamma=gamma, X=X[,-(2:3)], W=W, dist=my.dist, logDisp=logDisp, offset=Offy)
```

`stability.regimix` *Diagnostic checks to see if RCP groups are stable*

Description

For increasing size of hold-out samples, cooks distance and predictive log-likelihood are calculated and optionally plotted.

Usage

```
stability.regimix(model, oosSizeRange = NULL, times = model$n,
                        mc.cores = 1, quiet = FALSE, doPlot=TRUE)
```

Arguments

`model`	a regmix model, as obtained by the function `regimix`. This is the model whose stability is assessed. Model must contain titbits (see ?regimix and particular attention to the argument titbits=TRUE)
`oosSizeRange`	the size of the (successive) hold-out samples. If NULL (default), then a sequence of 10 sizes, from 1 to 0.2*model$n is used. The more numbers in this range, the slower the function will run.
`times`	the number of hold-out samples to use. If times=model$n and oosSize is 1, then the sample contains each and every site. Otherwise, it is a sample of size times from the possible combinations of possible hold-out sets.
`mc.cores`	the number of cores to farm the jobs out to
`quiet`	should the progress bar be displayed (bar for each oosSizeRange)
`doPlot`	should the plots be produced? Default is that they should be.

Details

The plots produced are: 1) leave-some-out Cook's distance (see `cooks.distance.regimix`) against holdout sample size; and 2) the predictive log-likelihood for times sites, against the holdout sample size. In both plots, the values from the original model have been added to the plot.

Value

`stability.regimix` produces a registab object. This is a list with the oosSizeRnage, disty (the mean Cook's Distance for each subset size), nRCP, n, predlogls (log-likelihood of out-of-sample sites), logl.sites (the in-sample log-likelihood for full data set).

See Also

`regimix`, `cooks.distance.regimix`, `plot.registab`

Examples

```
## Not run:
#not run as R CMD check complains about the time taken.
#This code will take a little while to run (about 3.5minutes on my computer)
system.time({
example( regimix);
stability.regimix( fm, oosSizeRange=seq( from=1,to=fm$n%/%5,length=5),
      times=fm$n, mc.cores=2, doPlot=FALSE);
})

## End(Not run)
```

`summary.regimix` *Summarises a regimix object.*

Description

A summary from a regimix object. It may be useful for something.

Arguments

`object`	a object obtained from fitting a RCP (for region of common profile) mixture model. Such as that generated from a call to regimix(qv).
`...`	ignored

Details

A table is printed that contains the coefficient values, their standard errors, and their z-statistic. The second and thrid columns may be unreliable for some parameters.

Method

summary(object, ...)

Author(s)

Scott D. Foster

`vcov.regimix` *Variance matrix for a regimix object.*

Description

Calculates variance-covariance matrix from a regimix object.

Arguments

`object`	an object obtained from fitting a RCP (for region of common profile) mixture model. Such as that generated from a call to regimix(qv).
`...`	ignored
`object2`	an object of class `regiboot` containing bootstrap samples of the parameter estimates (see regiboot(qv)). If NULL (default) the bootstrapping is performed from within the vcov function. If not null, then the vcov estimate is obtained from these bootstrap samples.
`method`	the method to calculate the variance-covariance matrix. Options are:'FiniteDifference' (default), `BayesBoot`, `SimpleBoot`, and `EmpiricalInfo`. The two bootstrap methods (`BayesBoot` and `SimpleBoot`, see regiboot(qv)) should be more general and may possibly be more robust. The `EmpiricalInfo` method implements an empirical estimate of the Fisher information matrix, I can not recommend it however. It seems to behave poorly, even in well behaved simulations. It is computationally thrifty though.
`nboot`	the number of bootstrap samples to take for the bootstrap estimation. Argument is ignored if !method %in% c(`FiniteDifference`,'EmpiricalInfo').
`mc.cores`	the number of cores to distrbute the calculations on. Default is 4. Set to 1 if the computer is running Windows (as it cannot handle forking – see mclapply(qv)). Ignored if method=='EmpiricalInfo'.
`D.accuracy`	The number of finite difference points used in the numerical approximation to the observed information matrix. Ignored if method != `FiniteDifference`. Options are 2 (default) and 4.

Details

If method is `FiniteDifference`, then the estimates variance matrix is based on a finite difference approximation to the observed information matrix.

If method is either "BayesBoot" or "SimpleBoot", then the estimated variance matrix is calculated from bootstrap samples of the parameter estimates. See Foster et al (in prep) for details of how the bootstrapping is actually done, and regiboot(qv) for its implementation.

Value

A square matrix of size equal to the number of parameters. It contains the variance matrix of the parameter estimates.

Method

vcov(object, ..., object2=NULL, method='FiniteDifference', nboot=1000, mc.cores=1, D.accuracy=2)

Author(s)

Scott D. Foster

References

Foster, S.D., Givens, G.H., Dornan, G.J., Dunstan, P.K. and Darnell, R. (2013) Modelling Regions of Common Profiles Using Biological and Environmental Data. Environmetrics *24*: 489–499. DOI: 10.1002/env.2245

Foster, S.D., Lyons, M. and Hill, N. (in prep.) Ecological Groupings of Sample Sites in the presence of sampling artefacts.

Index